Course Type	Course Code	Name of Course	L	T	P	Credit
DC7	CSC209	**Operating Systems**	3	0	0	9

Course Objective

This syllabus is designed in such a manner that it will provide the basic and fundamental knowledge on Operating Systems. The proposed syllabus is designed to cover Operating Systems in detail to provide better research and industry oriented understanding for UG students.

Learning Outcomes

On successful completion of this unit students will be able to:

- Identify the basic concept and describe the main responsibilities of a contemporary operating system (OS) and to explain the history leading to their current form.
- recognize and give examples of conflicting goals and compromises necessary in implementing an OS and configuring its run-time parameters
- identify and list application scenarios in which it is useful to use multiple threads of execution (including the fundamental need for multitasking in an OS)
- explain the concept of a process and the process control block (PCB) in a typical OS; recognize a PCB upon seeing the C code of such, and assess whether such a data structure contains everything that is necessary to handle the main tasks of a modern OS
- Provide a useful definition for a real-time system; give examples of actual real-time systems
- Understand how we can apply operating system concepts in industry

Unit No.	Topics to be Covered	Lecture Hours	Learning Outcome
1	Introduction, Categories of Operating Systems, Computer System Architecture, Interrupts, Storage Structure, Hardware Protection; OS Structures: OS Components;	4	Recognize and give examples of conflicting goals and compromises necessary in implementing an OS and configuring its run-time parameters
2	System Calls, System Structures, Virtual Machines, System Design Goal, SYSGEN	4	know and identify (from content description or C code), the most common data structures required in an OS implementation
3	Process Management: Process Concept, Process Sate, PCB, Process Scheduling, Schedulers, Process Creation, Process Termination, Co-operating Process, Producer Consumer Problem, Inter-process Communication, Client Server Communication, Threads, Process Synchronization, Critical Section Problem, Bakery Algorithms, Semaphores, Reader's Writer's Problem, Dining Philosopher's Problem;	5	Remember the most elementary challenges in concurrent programming (i.e., situations requiring mutual exclusion and synchronization) and solve them using semaphores (as defined by the POSIX threading interface). verify whether a given C (or similar pseudocode) program correctly solves the producer-consumer problem using multi-valued semaphores
4	CPU Scheduling: CPU Scheduler, Scheduling Criteria, Scheduling Algorithms: FCFS, SJF, Priority Scheduling, Round Robin Scheduling, Multilevel Queue Scheduling, Multilevel Feedback Queue Scheduling;	6	List and explain simple scheduling algorithms and give examples of applications in which each scheduler could be more beneficial than the others; likewise, choose the most suitable scheduling algorithm from a number of given choices, given an application scenario
5	Deadlock: Introduction, Deadlock Prevention, Deadlock Avoidance, Resource Allocation Graph Algorithms, Deadlock Detection, Prevention and Recovery;	6	Provide a concrete example (in C or in some pseudocode) of code that can lead to deadlock or data corruption due to a race; likewise, the student is able to tell whether a given code example (in C or similar pseudocode) has a bug that makes deadlock or data corruption likely to occur
6	Memory Management: Memory Hierarchy, Memory Types, Main Memory Architecture, Cache Memory, Address Binding, Dynamic Loading, Linking, Overlays, Logical vs Physical Addresses, Swapping, Contiguous Memory allocation, Fragmentation, Segmentation;	7	Know what the principle of locality stands for, how it is used in a typical memory system, and how the principle can be used in applications other than computer technology and OSs. translate a virtual memory address into a physical address, given a page table (of a given simple "toy" computer with very tiny address space); understand and explain how a shared memory area can be implemented using VM addresses in different processes

7	Virtual Memory, Paging, Demand Paging, Page Replacement Algorithms, Thrashing;	5	Describe how the page fault exception is handled when the reason for fault is a reference to an existing but swapped-out page, and the LRU page replacement algorithm is selected
8	Secondary Storage Structure: Disk Structure, Disk Scheduling, Disk Management; Case study: Unix and DOS;	5	Understand and explain how a shared memory area can be implemented using VM addresses in different processes

Text Books:

1. Abraham Silberschatz, Peter B. Galvin, Greg Gagne, Operating System Concepts, 9th Edition, Wiley Global Education, 2012.

Reference Books:

William Stallings, Operating Systems: Internals and Design Principles, GOAL Series, Pearson international edition, 2009.